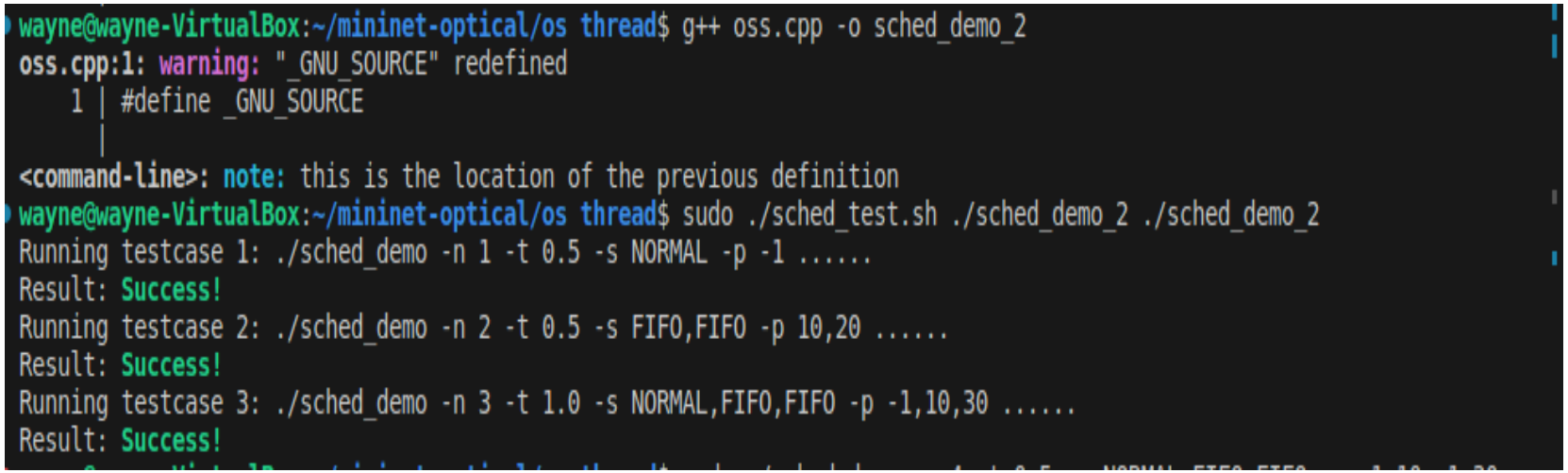
圖下為我創立oss.cpp檔經過g++編譯產生執行檔,並且經由老師的test檔通過測試3個success



Q1:My program detail

我的程式中:

1.先吃入所有函式庫與所有的define

Void{

1.設定了全域變數barrier使得所有程式能一起等待同步執行thread

2.利用*thread_inf吃入structure中所需的所有參數(id,policy..)

3.計算執行續需要時間,並換成毫秒

4.在主程式中,使每一條thread運行時印出自己,並以busy waiting等待

如果超過指定時間就跳出while迴圈釋放cpu與其他人爭奪cpu

}

我的程式中:

Main{

1.先解析所有命令參數

2.創建多個thread(包含優先權及排程之類的空間)

3.設定cpu affinity

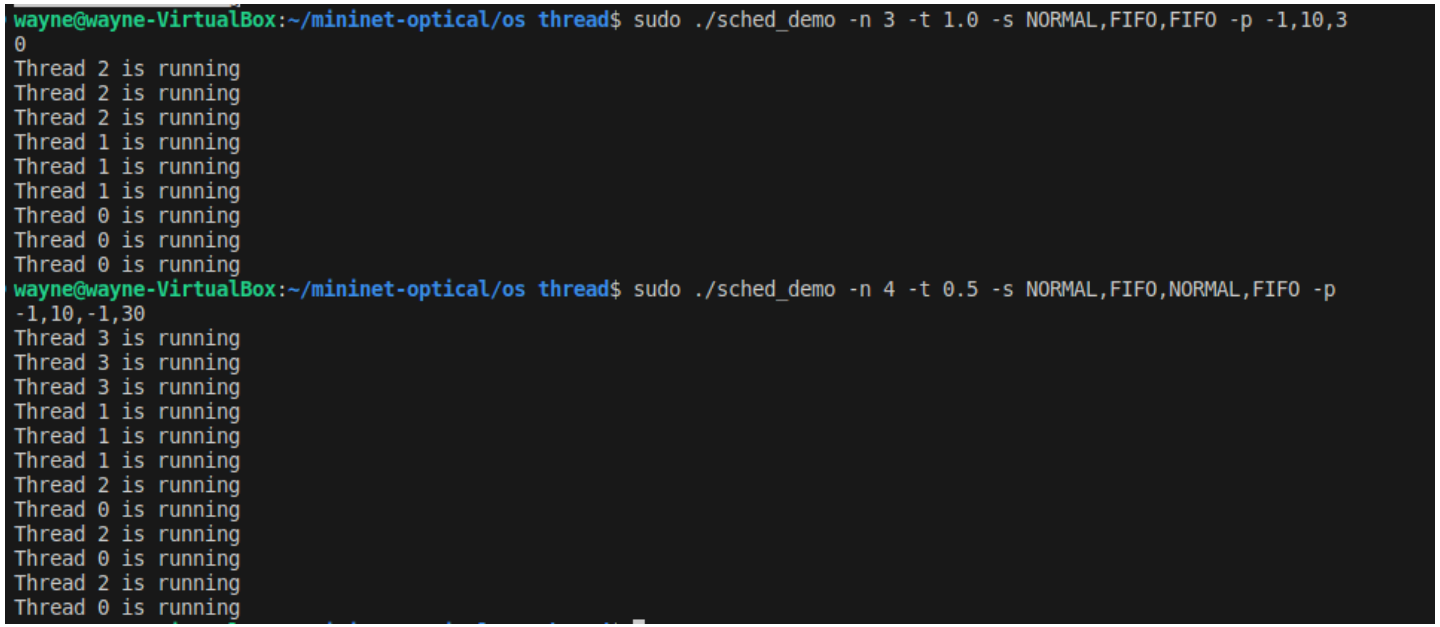
4.設定每一個執行緒的屬性(等等可以use &attr使create時吃到)

5.藉由barrier同時執行接著等待完成

}

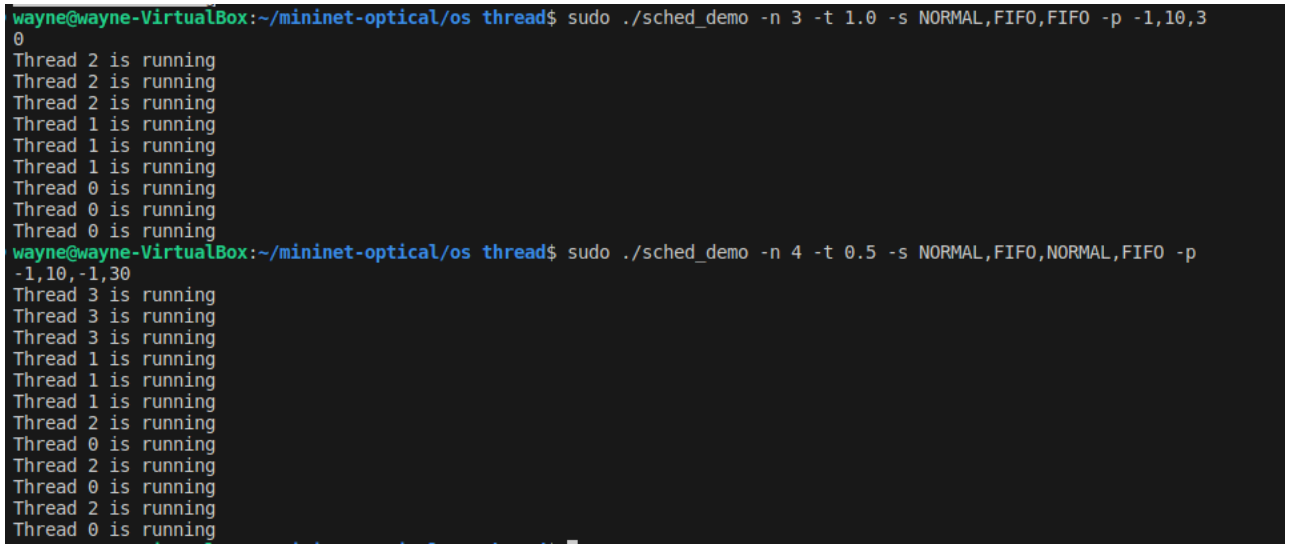
Q2:./sched_demo -n 3 -t 1.0 -s NORMAL,FIFO,FIFO -p -1,10,30 之Result

因設定的三個thread分別為Normal,FIFO,FIFO
所以在即時排序下FIFO優先,又第三個thread之priority為30較大,所以thread2先執行直到結束
再來thread1為FIFO優先,所以thread1執行並印出3次
最後為thread0執行3次



Q3:./sched_demo -n 4 -t 0.5 -s NORMAL,FIFO,NORMAL,FIFO -p -1,10,-1,30之resukl之result

因設定的四個thread分別為Normal,FIFO,Normal,FIFO
所以在即時排序下FIFO優先,又第四個thread之priority為30較大,所以thread3先執行直到結束
再來thread1為FIFO優先,所以thread1執行並印出3次
最後為thread0和thread2為normal,沒特別設定priority(皆為-1)
所以每一次busy waiting時間到釋放cpu,兩個相等優先權的thread皆有可能拿到cpu
因此thread0,2交替出現



Q4:implement n-second-busy-waiting

我有代入n=0.5與n=1結果差不多

因題目說不可使用sleep,我使用的方式是使thread在while迴圈一直處於running狀態直到時間到達我們的設定值,break跳出while迴圈與大家重新爭奪cpu